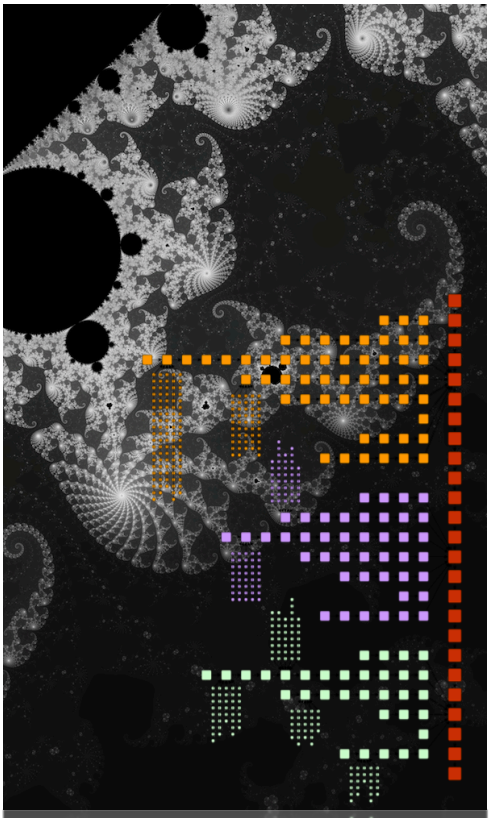


ASSETREE

Distributed Fractal Ledger

ABSTRACT

ASSETREE: 基于 DFL 技术构建一种高并发、可扩展、分区的分布式账本体系，打造连接现实世界海量真实资产到虚拟世界的区块链网络

Z. Johnson

Tech. Whitepaper For SharesChain

目录

一、 前言

1.1 单链式区块链

比特币区块链

消耗大量能源。比特币区块链系统的工作量证明共识机制需要消耗大量计算资源和能源，而且还在快速增长中。

民主特征淡化。比特币区块链要求 50%以上的计算资源掌握在诚实用户的手中，以保证系统的一致性运转，但中本聪应该没有预见到比特币区块链中的计算资源集中在少数大型矿池中，使比特币区块链偏离了其早期宣称的民主特征。从比特币历史上关于扩容的讨论以及多次分叉证明比特币区块链已经形成了中心化程度很高的社区结构。

交易速度极低。一个比特币区块的大小为 1M，大约能容纳 2000 笔左右交易，因为平均每 10 分钟产生一个区块，比特币平均每秒钟能支持 3-7 笔交易。

以太坊区块链

交易速度受限。以太坊目前的性能（7-15TPS）在面对任何具有规模的场景时都会显得极其单薄。

缺乏并发机制。以太坊是分布式的计算网络，负责全球基于以太坊的项目的分布式计算，其单线程模式的单链设计导致稍有规模的计算请求就会被排队阻塞，进一步导致单个以太坊应用（如以太坊养猫）影响全球所有其他以太坊应用的计算，作为一个通用的计算网络，并发机制严重缺乏。

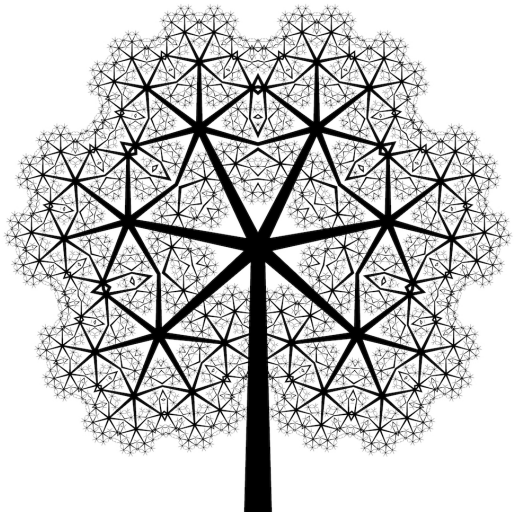
1.2 DAG 式区块链

DAG 高速异步区块链技术，但存在不少挑战，比如对全网交易状态的强一致性没有严格的要求；在通过无子单元获得并发验证的同时又要控制无子单元的收敛程度避免过多的未被确认的交易；在构建主链的同时又要支持足够多的子链来获得并发能力；虽然去除了区块链的概念，当子链数量较少时就可能退化为类似单链的效率。

其实，任何分布式账本体系都有自身的定位，包括采用了 DAG 技术的体系，如 IOTA 虽牺牲了全网交易的强一致性，但在物联网方面的适应能力就很出色[1]，Byteball 则注重支付领域的业务。**Assetree 的定位就是希望在不失去良好一致性的同时获得高速并行能力，支持现实世界海量真实资产应用的有效运行。**

1.3 分形账本结构

世界是非线性的，分形无处不在，复杂的结构往往可以由简单的单元构成，**我们试图构建一种一致性强、应用隔离、高并发的分布式账本体系，用于资产的登记、管理、和流转，而树结构区块链（我们称之为 Assetree）**是一个不错的选择：逻辑结



构稳定、清晰，十分有利于基于地址空间的分区设计；链式结构已经被证明可以拥有完美的分布式一致性解决方案；树结构区块链可以支持分形结构的扩展支持高维设计：分布式分形账本（DFL：Distributed Fractal Ledger）。

DFL 强调基础结构的简单性、基础结构的可连接性、连接规则的可递归性。不同的基础结构，在不同的随机分布下，可以构建出不同的 DFL，取得不一样的访问路径和效率。Assetree 类似大自然中的一棵树，树的每一部分都概率满足分形的结构特点，反过来，我们只需定义好基础结构的规则，则可以递归复制出一棵树来管理所有账本数据。

一致性问题是分布式计算最为基础的问题，是指对于分布式网络中的节点，给定一系列操作，在约定协议的保障下，达到对处理结果的共同认同。达到认同的过程可以用共识算法来进行表述，同时数据的结构和状态会严重影响共识算法的表现。由于 CAP 原理（Eric Brewer 2000 年 ACM 会议提出的分布式计算领域重要原理之一）的存在，即分布式系统不能同时满足一致性（Consistency）、可用性（Availability）和分区容忍性（Partition），系统设计时需要有所取舍。**我们的倾向是尽力去除系统的不确定性，哪怕牺牲一定的效率和空间。同时，通过对地址空间进行分区管理，根据可能的节点数和并发量，我们可以设计递归构建账本的结构的层级，以及地址空间划分的程度。**

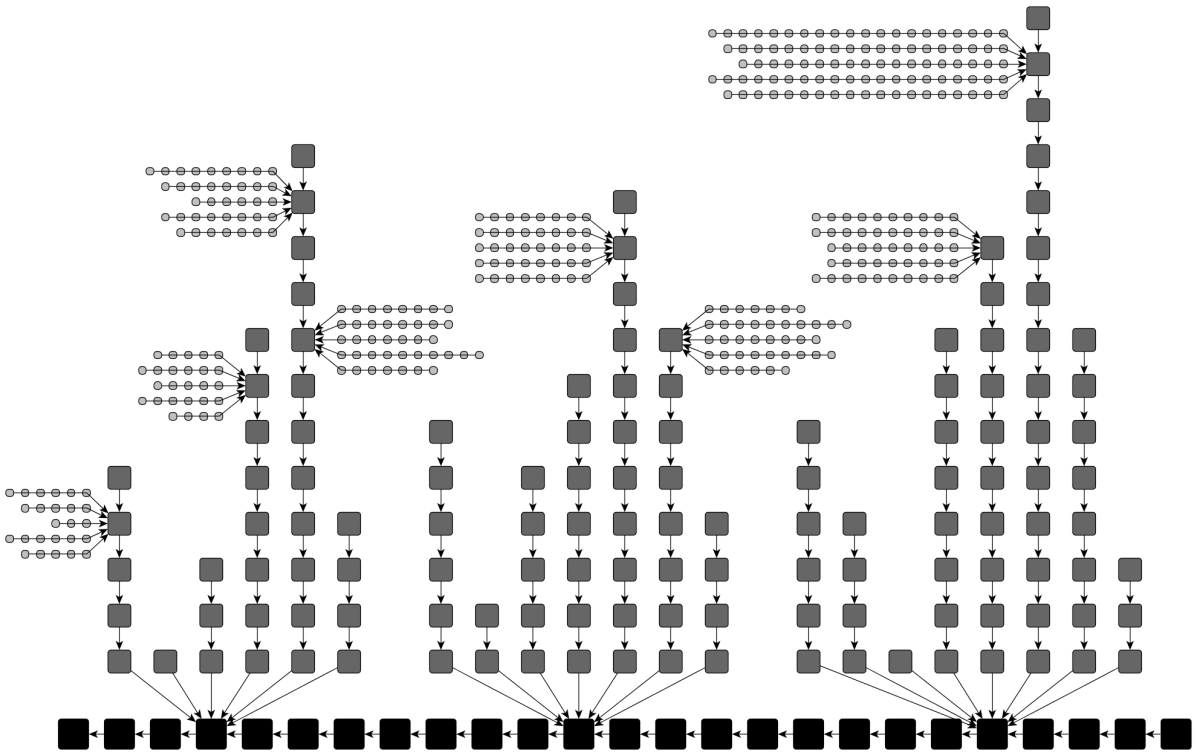


图 1 Assetree 账本结构示意图

1.4 智能合约应用

如果把底层区块链比作数据层的话，智能合约就归属于业务层，Assetree 提倡数据层和业务层的解耦合设计；区块链数据的存储应该使用专业的区块数据库（BlockDB）来读写和索引；智能合约的处理应该交由智能预言机（Smart Oracle Server）来处理，同时获得和区块链体系外的互联网进行交互的能力；**Assetree 将支持资产在网络中进行交易时自带智能合约**，使得资产交易双方满足发行方等因素的限制，或者说增强发行方等对资产的编程和控制能力。

1.5 连接真实资产

海量真实资产很难依靠单一机制保障速度和可扩展性，Assetree：支持资产发行时对子链共识算法和参数定义等的配置；**设计一种应用隔离的分布式账本体系，避免单个应用对其他应用的影响，使得多个应用可以并发运行**；支持地址分区的账本体系，获得应用内的并发能力。

二、ASSETREE 网络结构

- Assetree 由负责资产发行的主链和负责资产交易的子链构成；

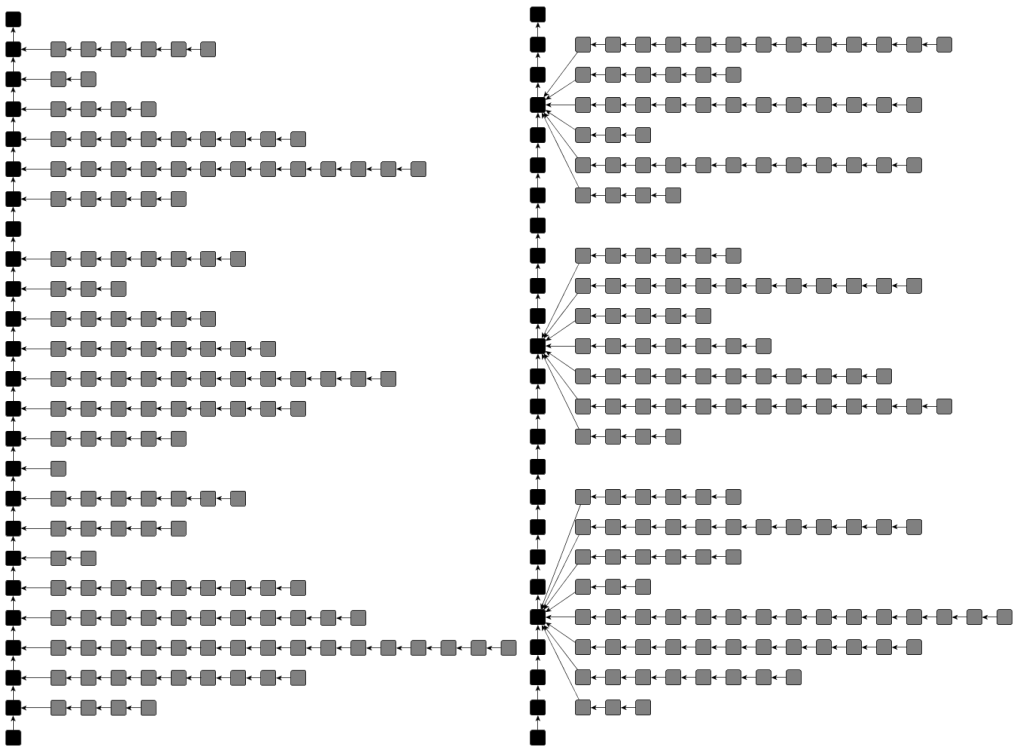


图 2 Assetree 子链示意图

- 主链（MainChain、MC），如图 1 所示 X 轴方向的深黑色链，主链有且只有一条；子链（SubChain、SC），主链以外的其他链即为子链，每一

条子链的第一个区块都唯一指向主链上的一个区块（GenesisBlock、GB），如图 2 所示；

- 每个应用可以有一条或者**多条子链**，按账户地址取模来区分，比如要分成 16 条子链，那么支付方用户地址模 16 等于 0 的所有交易形成区块链接在#0 子链上；子链数目最大值为 100 万；
- Assetree 利用 Merkle 树快速校验交易数据是否被修改；
- Assetree 采用 UTXO 的交易记录模型，但是严格限制资金来源都属于同一个支付方，**限制输出资金的范围为支付方和某一个子链空间内的若干接收方地址**；针对多方支付和多方接受并且跨>=3 链的情况，上层应用要负责规避或者拆分成多个 2 链交易。
- Assetree 考虑支持一种新的交易模型：切分 UTXO 成只保留资产接受方信息的 ITXO 和只保留资产支付方信息的 OTXO，并分开打包 ITXO 和 OTXO 到对应的子链上，**断开交易双方的关联性，使得交易历史不可跟踪**；
- Assetree 设计支持三种类型的节点：超级节点，同步拥有所有账本数据，可支持所有类型资产的交易确认和共识，以及智能预言机的服务；全节点，同步拥有某一类资产的所有账本数据，负载低，可支持单一资产的确认和共识，以及智能预言机的服务；轻节点，同步拥有某一类资产的和自己地址相关子链的部分账本数据，适合资产的管理和交易；

三、ASSETREE 资产发行

- Assetree 发行原生 Gas Token 用于激励自身区块链网络的建设：SCTK；**SCTK 自己定义子链的数量为 100 万条，按单链平均 1000TPS 计算，设计未来可支持 10 亿的并发量；**
- 应用的资产发行信息形成区块（GB）记录在主链上；信息包括资产名称、ID、发行总量、最小单位、是否增发、是否支持挖矿铸币、是否支持销毁、子链数目、共识算法、共识节点随机选中率 R、发行协议、资产自带智能合约等等一系列配置信息。每个应用发行的 Token 是不一样的。
- **应用方发行资产需支付一定数量的 SCTK，和子链数量成正比，如果应用 Token 发行成功，此 SCTK 会被网络销毁。**

四、ASSETREE 资产管理

Assetree 致力于为上层资产登记、管理和流转业务逻辑提供丰富的操作接口和服务。比如资产的多账户管理，权限配置，信息发布，点对点通信，以及数据加密存储等。

- **支持账户间资产操作，包括冻结、分期、质押、销毁等操作；**

- **支持多角色账户地址，实现多账户签名对资产的交易控制和多账户对信息修改的控制；**
- **支持账户间通信信道，实现账户间加密、非加密实时可信通信；**
- **支持账户数据的管理，实现账户数据机密、非加密存储和修改；**

五、ASSETREE 资产交易

- SCTK 的交易信息形成区块链记录在 SCTK 自己的子链上；SCTK 的交易需要支付一定数量的 SCTK，交易完成，此部分 SCTK 会被销毁，同时矿工将获得一定数量的 SCTK 奖励。
- 每一类资产有自己单独的子链来承载所有发行后的分布式计算；应用的资产交易信息形成区块记录在子链上；
- 用户发起资产交易时需支付一定数量的应用 Token 作为手续费，支付得越多，被矿工打包的可能性越大；

六、ASSETREE 共识机制

在公有链中的共识机制一般是工作量证明机制 POW 或权益证明机制 POS，节点对共识形成的影响力直接取决于它们在网络中所拥有的资源占比。**通过随机算法，可以避免共识局限在少数节点或矿池上**，避免诸如白名单攻击、需要信任陌生主体、选票人为干预等问题，大大加强整个体系的民主化，同时，Assetree 默认采用类似 POS 的共识机制来避免挖矿消耗巨大的能量。

Assetree 网络根据消息的类型（交易、通信、存储等）维护若干请求消息队列（Message Queue），作为共识节点的超级节点和全节点支持多线程监听、处理消息，每一个线程必须配置一个和其所监听的子链地址空间对应的 SCTK 账户地址 ADR，用于参与共识时的费用计算。

针对交易消息，监听线程有两种，一种是监听处理跨子链的交易，Double-Thread，如线程 DT1 监听#X 和#Y 两条子链上的交易，即所有发生在#X 和#Y 之间的交易都会被 DT1 监听到，同时，这类线程天然可以处理不跨子链的交易；另外一种是监听处理单个子链内的交易，Single-Thread（这类交易轻节点都可以参与共识计算）。此外，通过给 DT、ST 配置多个链对内地址，可以让线程获得监听多个链对内交易的能力，但是此线程不能并发处理多个链对，当然，可以启动多个线程来获得对链对的并发处理能力。

一个跨子链的交易消息，如果涉及>=3 链的交易时会被业务层分割为多个两链间交易的消息，当交易消息到来时，至少有两种选择来进行共识处理：其一是参与共识的节点中的一个 DT 线程同时负责了相关两个子链的共识处理，而且是捆绑的，如果一个失败，也意味着另外一个共识失败，这就避免了交易信息在两个子链上的状态的不一致，当然，交易信息不会被重复计算，因为特定用户的所有交易信息只需统计分析单一子链账本即可。

Assetree 通过随机种子（如子链最新区块的哈希值）的选取保证随机性和民主性：和 DPOS 不同，不是权益代理，而是随机选出共识节点，再根据权益大小来确定记账资格，我们姑且称这种机制为 RPOS（Random POS）。假设子链最新的区块的哈

希为 Hash(HDR)，想参与共识计算的节点（候选者，线程账户地址为 ADR）需要配置自己监听的子链对（#X#Y、或者#X#X）信息，每一个候选者都质押一定数量的 SCTK 作为记账的资格证明，那么 Hash(HDR) + Hash(ADR) = (INT)H，如果 H mod 1/R = 0，那么本用户被选中成为见证者（否则不再参与计算），然后，假设 IDX 为选举轮次，H=(INT)Hash("H+IDX")，在见证者中再次选举，直到被选中的见证者数不超过 C（可配置，如 17）个，其中质押 SCTK 最多的见证者组装区块并和其他见证者通过 PBFT 算法进行快速签名和共识。RPOS 节点选取过程不需要网络中候选节点进行复杂的计算和大量通信，每个节点使用既定的计算方法即可算出自己是否被选中（随机因子的选取将进一步调优避免重复选中），其他节点也可进行验证。共识成功，见证者按质押 SCTK 比例分享交易费用和网络奖励，否则销毁记账人的质押 SCTK。

网络奖励 SCTK 的量被设计为随着时间的推移而逐年减少，未来见证者的收益主要来自于所监听的子链对应的应用 Token（交易手续费），目的是加大竞争力度，让优秀的应用得到网络的高度认可。见证者 SCTK 的自动销毁和原生奖励都会以特殊交易的形式广播给超级节点，由超级节点来批量确认、共识、记账到 SCTK 对应子链上，实现信息的一致性。

Assetree 前期计划使用的共识算法是 RPOS+PBFT，预期子链上 1000TPS 的处理能力，同时，Assetree 提供共识算法的扩展接口，使得区块组装者可以选择更加高效的算法，参考[2]。

共识节点根据自己的类型（超级节点、全节点和轻节点）可以设置若干监听线程（ST、DT）并发的参与全网共识计算，但链对（#X#Y）上的共识计算是线性的。理论上，在交易随机均匀分布的前提下，共识节点越多，全网支撑的并发量越大。

七、智能合约

Assetree 支持图灵完备的智能合约体系，设计支持智能预言机（Smart Oracle）。智能合约是业务层的协议，底层区块链是数据层的协议，这两种协议解耦合设计是 Assetree 追求的方向，以支撑上层业务的复杂性。

八、其他说明

本文提到的各种参数、数值，在后续数学模拟验证过程中会进行科学调整。

如需支持轻节点对应用交易的共识，可以人为的使用属于某一个特定子链地址空间的账户来进行资产的相关操作，即人为保证不发生跨链交易，这对小型应用、智能终端来说是一个非常不错的选择；其实，三类节点之间没有严格的边界，完全在于计算机性能、网络速率和人为配置等的制约。

DFL 是一种设计模式，有其自身的适应性，根据具体的业务场景可以设计出不同的 DFL 体系，也会得到不同的计算复杂度，如共识算法，账户资产统计、寻址管理等等，更高级别的分形扩展可以考虑地址空间的组合，一种是取全局更大的地址空间，一种是叠加小空间。一般我们不需要担心地址空间不够用，全宇宙的原子为10^{80}数量级个，一个 40 位的地址拥有10^{48}数量级个，64 位10^{77}数量级，在马斯克坐到火

星办公室里前我们可能不需要考虑空间的问题。DFL 非常适合空间划分，按地址空间划分和管理交易最大的优势在于大大降低了类似 DAG 技术中的概率不确定性。

九、参考资料

[1] https://iota.org/IOTA_Whitepaper.pdf
[2] https://arxiv.org/pdf/1607.01341.pdf
[3] https://github.com/ethereum/EIPs/issues/650
[4] http://pmg.csail.mit.edu/papers/osdi99.pdf
[5] http://ethfans.org/posts/Sharding-FAQ
[6] http://ethfans.org/posts/ethereum-sharding-and-finality
[7] https://byteball.org/Byteball.pdf
[8] https://bitcoin.org/bitcoin.pdf
[9] https://en.bitcoin.it/wiki/Atomic_crosschain_trading
[10] https://github.com/bitcoin/bitcoin
[11] https://trustnote.org/TrustNote-WhitePaper-en.pdf
[12] https://github.com/iotaledger
[13] https://github.com/EOSIO/Documentation